カーネル法入門
5. カーネル法のその他の話題

福水健次

統計数理研究所／総合研究大学院大学

大阪大学大学院基礎工学研究科・集中講義

2014 September

- 効率的計算
  低ランク近似の方法

- 構造化データ
  非ベクトルデータに対するカーネル

カーネル法の計算効率化

グラム行列計算

– カーネル法の計算： グラム行列による線形代数演算
  データ数のサイズの行列

  • 元の空間の次元が高くても計算量の問題は（あまり）生じない

  • データ数が大きいと計算量の問題が生じる
    逆行列計算，固有値計算 $O(n^3)$ in time

■ 計算効率化への一般的なアプローチ

- 低ランク近似によるGram行列の近似
  - 不完全Cholesky分解
  - Nyström近似

- ランダムなカーネル展開
  - Random kitchen sink

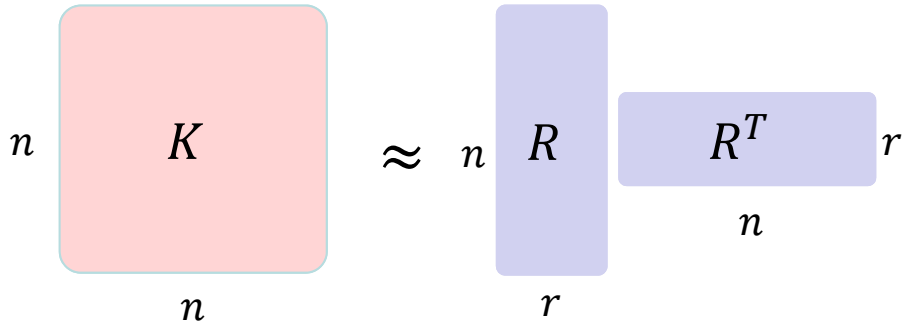
- 少数データによる表現
  - データのランダムサンプリング
  - Core Vector Machine（Tsang et al 2005）Core set
  - Kernel herding（Chap 6で扱う）

上記とは別に，個々のアルゴリズムの効率化はさまざまに検討されている.

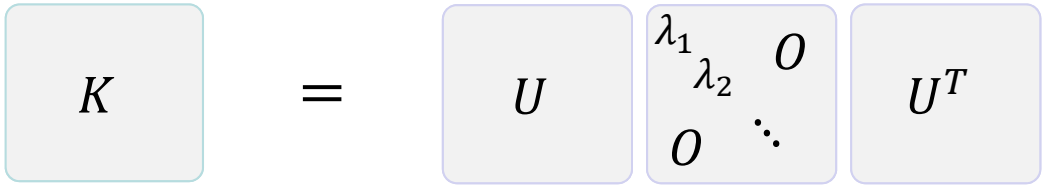
低ランク近似

– 低ランク近似

$$K \approx RR^T, \qquad R\colon n \times r \text{ 行列 } (r \ll n)$$



c.f. 固有分解



– ランク r はあまり大きくなくてよい.

典型的な例で, グラム行列の固有値の減衰は早いことが知られている. (Widom 1963, 1964; Bach & Jordan 2002).

■ 計算効率化の例

– カーネルリッジ回帰

$$f(x) = Y^T(K_X + \lambda I_n)^{-1}\mathbf{k}(x) \qquad \text{time : } O(n^3)$$

低ランク近似: $K_X \approx RR^T$.

Woodburyの公式を用いると

$$Y^T(K_X + \lambda I_n)^{-1}\mathbf{k}(x) \approx Y^T(RR^T + \lambda I_n)^{-1}\mathbf{k}(x)$$

$$= \frac{1}{\lambda}\{Y^T\mathbf{k}(x) - Y^T R(R^T R + \lambda I_r)^{-1}R^T\mathbf{k}(x)\}$$

time : $O(r^2 n + r^3)$

Woodburyの公式

定理５．１　Woodbury (Sherman–Morrison–Woodbury) の公式

$A: n \times n$可逆行列, $U: n \times r$行列, $V: r \times n$行列

$$(A + UV)^{-1} = A^{-1} - A^{-1}U(I + VA^{-1}U)^{-1}VA^{-1}.$$

証明）　直接計算.

特に$r = 1$のとき,

$$(A + uv^T)^{-1} = A^{-1} - \frac{1}{1 + v^T A^{-1} u} A^{-1} u v^T A^{-1}.$$

復習：Cholesky分解

A: $n \times n$ 半正定値行列.

A のCholesky分解:　$A = RR^T$

R: 下半三角行列

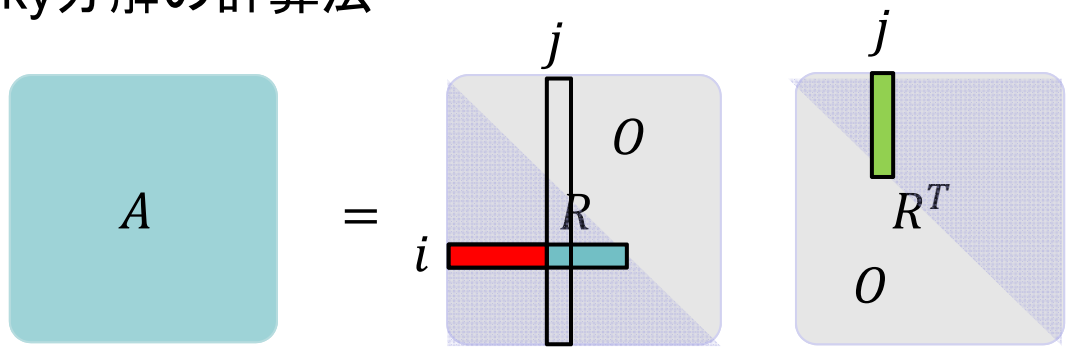
– Fact: rank$A = r$ のとき，ある置換行列 P があって，

$$PAP = RR^T, \qquad R = \begin{bmatrix} R_{11} & O \\ R_{21} & O \end{bmatrix}$$

R_{11}: $r \times r$下半行列，対角成分は正

– 「掃き出し法」に相当する.

– Choelsky分解の計算法



$1 \le j \le i \le n$のとき

$$A_{ij} = \sum_{k=1}^{j} R_{ik}R_{jk} = R_{ij}R_{jj} + \sum_{k=1}^{j-1} R_{ik}R_{jk}$$

したがって

$$R_{ij} = \frac{A_{ij} - \sum_{k=1}^{j-1} R_{ik}R_{jk}}{R_{jj}} \qquad (j+1 \le i \le n)$$

R_{jj}を決めると，R_{ij} $(j+1 \le i \le n)$は$R_{*k}(k < j)$により決まる.

→ 列に関する逐次計算が可能

不完全Cholesky

– Cholesky分解を途中まで行う.

– 近似誤差の評価が可能: $\mathrm{Tr}[A - RR^T] < \epsilon$

[アルゴリズム]

A: $n \times n$半正定値行列, ϵ:しきい値

1. [初期化] $i = 1$. R :=Null, $A' = A$, $P = I_n$. $R_{jj} = A_{jj}$ $(1 \le j \le n)$.
2. If $\sum_{j=i}^{n} R_{jj} < \epsilon$, END. Otherwise, go to 3.
3. $j^* := \arg\max_{j=i,\ldots,n} R_{jj}$
4. [置換] $A'_{[ij^*],:} = A'_{[j^*i],:}$, $A'_{:,[ij^*]} = A'_{:,[j^*i]}$, $R_{[ij^*],1:i} = R_{[j^*i],:1:i}$,
   $P_{ii} = P_{j^*j^*} = 0, P_{ij^*} = P_{j^*i} = 0.$
5. $R_{ii} = \sqrt{A'_{ii}}$
6. [第i列の計算]

$$R_{i+1:n,i} = \frac{A'_{i+1:n,i} - \sum_{k=1}^{j-1} R_{i+1:n,k} R_{ik}}{R_{jj}}$$

7. [対角成分の更新] $R_{jj} := A'_{jj} - \sum_{k=1}^{i} R_{jk}^2 \quad (i+1 \le j \le n)$
8. $i := i + 1$ and go to 2.

Output: $n \times (i-1)$行列 R, 置換行列 P

■ 計算量

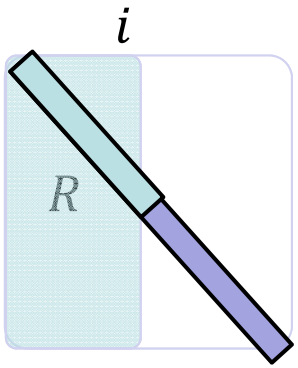
Rの列数が r とすると,

- 時間： ステップ6で, $O(ni)$. 合計 $O(nr^2)$.
- メモリー： $O(nr)$ （注: R_{jj}は別配列にしておく）
- 第i列の計算には, Aの第i列と対角成分しか使っていない.

→ Gram行列を最初にすべて計算する必要はない.
使っているGram行列の要素数も $O(nr)$.

■ 近似誤差

- R_{jj} $(i+1 \leq j \leq n)$ に $A - R^i R^{i^T}$ の対角成分を格納（ステップ7）.



Nyström近似

- Nyström近似は，もともと積分作用素の固有関数・固有値の近似手法として知られていた.
- Williams & Seeger (2001) がGram行列近似に応用した.

- 固有値問題: kは正定値カーネル

$$\int k(y,x)\phi(x)dP(x) = \lambda\phi(y), \qquad \int \phi(x)^2 dP(x) = 1$$

固有値 $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$, 対応する固有ベクトル $\phi_1, \phi_2, \ldots$

- $X_1, X_2, \ldots, X_m \sim P$, i.i.d. による近似

$$\frac{1}{m}\sum_i^m k(y,X_i)\phi(X_i) \approx \lambda\phi(y), \qquad \frac{1}{m}\sum_i^m \phi(X_i)^2 \approx 1.$$

$y = X_j$ とおくと

$$\frac{1}{m}\sum_{i}^{m} K_{ij}^{(m)} \phi(X_i) \approx \lambda \phi(X_j), \qquad \frac{1}{m}\sum_{i}^{m} \phi(X_i)^2 = 1.$$

$K^{(m)}$の固有分解: $K^{(m)} = U^{(m)} \Lambda^{(m)} U^{(m)^T}$, $\Lambda^{(m)} = \mathrm{Diag}(\lambda_1^{(m)}, \ldots, \lambda_m^{(m)})$

近似:

$$\phi_i(X_j) \approx \sqrt{m}\, U_{ji}^{(m)}, \qquad \lambda_i \approx \frac{\lambda_i^{(m)}}{m}$$

任意の y に対し,

$$\phi_i(y) \approx \frac{\sqrt{m}}{\lambda_i^{(m)}} \sum_{j=1}^{m} k(y, X_j) U_{ji}^{(m)} \qquad \cdots(*)$$

■ Gram行列の近似

- $P = \frac{1}{n}\sum_{i=1}^{n} \delta_{X_i}$　　$X_1, \ldots, X_n$: オリジナルのデータ
- $X_1, \ldots, X_m$: Pからの一様サンプル（簡単のため，添字ははじめのm個）

$m = n$のときが厳密解.

$$\phi_i(X_j) = \sqrt{n} U_{ji}^{(n)}, \qquad \lambda_i = \frac{\lambda_i^{(n)}}{n}$$

欲しいのは，$U_{ji}^{(n)}$, $\lambda_i^{(n)}$の近似．(＊)より，

$$U_{ji}^{(n)} \approx \sqrt{\frac{m}{n}} \frac{1}{\lambda_i^{(m)}} \sum_{a=1}^{m} K_{ja} U_{ai}^{(m)} \equiv \tilde{U}_{ji}^{(n)} \ (1 \le j \le n), \qquad \lambda_i^{(n)} \approx \frac{n}{m} \lambda_i^{(m)} \equiv \tilde{\lambda}_i^{(n)}$$

さらに，大きいp個の固有値のみを使うと

$$K \approx \sum_{i=1}^{p} \tilde{\lambda}_i^{(n)} \tilde{U}_i^{(n)} \tilde{U}_i^{(n)T} \equiv \tilde{K}_p$$　　Nyström近似

■ Nystrom近似の演算量

- $K^{(m)}$の固有分解： $O(m^3)$
- $\tilde{U}_i^{(n)}$の計算：各iにつき $O(nm)$

トータル　$O(m^3 + pnm)$

Random Kitchen Sink

– 非常に大きなデータ（10万～）などでも使うことを想定

– 復習： Bochnerの定理

$\mathbf{R}^m$上連続で平行移動不変なカーネル

$$k(x,y) = \int \exp\left(\sqrt{-1}\omega^T(x-y)\right) d\Lambda(\omega)$$

Λは非負測度なので，適当に正数倍することにより，
確率測度$\int d\Lambda(\omega) = 1$ に正規化しておく.

– Random Kitchen Sink (Rahimi & Recht 2008)

周波数領域でサンプリング

$$\omega_1, \ldots, \omega_L \sim \Lambda, \text{ i.i.d.}$$

$$k(x,y) \approx \frac{1}{L}\sum_{\ell=1}^{L} \exp\left(\sqrt{-1}\omega_\ell^T(x-y)\right)$$

$$k(x,y) \approx \frac{1}{L}\sum_{\ell=1}^{L} \exp\left(\sqrt{-1}\omega_\ell^T (x-y)\right)$$

カーネルが実数値とすると，

$$k(x,y) \approx \frac{1}{L}\sum_{\ell=1}^{L} \cos\left(\omega_\ell^T (x-y)\right)$$

$$= \frac{1}{L}\sum_{\ell=1}^{L} \{\cos\left(\omega_\ell^T x\right)\cos\left(\omega_\ell^T y\right) + \sin\left(\omega_\ell^T x\right)\sin\left(\omega_\ell^T y\right)\}$$

$$Z(x) := \frac{1}{\sqrt{L}}\left(\cos(\omega_1^T x), \ldots, \cos(\omega_L^T x), \sin(\omega_1^T x), \ldots, \sin(\omega_L^T x)\right)^T$$

とおいて

$$k(x,y) \approx Z^T(x)Z(y)$$

- $Z(x)$ を基底関数として，リッジ回帰などを行う ≈ カーネル法
  （$L \ll n$ の状況．グラム行列で表現しない）

■ 比較

– Rahimi & Recht 2008 では，Core Vector Machine よりもよい結果を得ている.

Dataset	Fourier+LS	Binning+LS	CVM	Exact SVM
CPU regression 6500 instances 21 dims	3.6% 20 secs $D = 300$	5.3% 3 mins $P = 350$	5.5% 51 secs	11% 31 secs ASVM
Census regression 18,000 instances 119 dims	5% 36 secs $D = 500$	7.5% 19 mins $P = 30$	8.8% 7.5 mins	9% 13 mins SVMTorch
Adult classification 32,000 instances 123 dims	14.9% 9 secs $D = 500$	15.3% 1.5 mins $P = 30$	14.8% 73 mins	15.1% 7 mins $\text{SVM}^{\text{light}}$
Forest Cover classification 522,000 instances 54 dims	11.6% 71 mins $D = 5000$	2.2% 25 mins $P = 50$	2.3% 7.5 hrs	2.2% 44 hrs libSVM
KDDCUP99 (see footnote) classification 4,900,000 instances 127 dims	7.3% 1.5 min $D = 50$	7.3% 35 mins $P = 10$	6.2% (18%) 1.4 secs (20 secs)	8.3% < 1 s SVM+sampling

– Random Kitchen sinkは，データを使わずに少数の基底をランダムに選ぶ　c.f. グラム行列の低ランク近似.

– Tsang et al 2005 では，CVMは不完全Choleskyよりも少ない演算量で同等の識別／回帰の性能を得ている.

構造化データ

複雑な構造を持つデータ（ストリング，ツリー，グラフ）
に対して定義されるカーネルとその計算法を紹介する

構造化データの処理

■ カーネルの利用

正定値カーネル $k(x, y)$ ： x, y はベクトルデータでなくてよい

– どんなデータでもOK
  - 長さの違うシンボル列　＝　ストリング
  - ツリー構造
  - グラフ表現されたデータ

– カーネル法　→ 非ベクトルデータのベクトル化
  カーネルが定義されると，SVM，カーネルPCA, などの利用が可能

– 計算すべきもの ＝ データに対するグラム行列　$k(x_i, x_j)$

ストリング

■ ストリング

- アルファベット Σ : 有限集合
- ストリング : Σ の要素の有限長の列
  - 例） $\Sigma = \{ a, b, c, d, \ldots, z \}$
    ストリング cat, head, computer, xyydyaa,…

□ Σ^p : 長さ p のストリング全体

□ Σ^* : 任意の長さのストリング全体 $\Sigma^* = \bigcup_{p=0}^{\infty} \Sigma^p$

注） $\Sigma^0 = \{\varepsilon\}$: 空ストリング

- 記号法

  s : $s_1 s_2 \ldots s_n$ ストリングに対し

  | s | ・・・ ストリング s の長さ = n

  s[i : j] ・・・ $s_i \ldots s_j$ という s の部分列

  s, t に対し結合 s t = $s_1 s_2 \ldots s_n t_1 t_2 \ldots t_m$

ストリングカーネル

■ ストリングカーネル

- Σ^* 上の定義された正定値カーネル ・・・ 2つのストリング s, t の類似度
  - 一致する部分列を数え上げるタイプが多い
  - 効率的な計算の工夫が重要 ・・・ 再帰式（漸化式)など
    Dynamical Programming（DP)

■ 典型的な応用先

- 自然言語処理
  - 文字列： $\Sigma = \{a, b, c, \ldots, z\}$
  - 単語列： $\Sigma = \{$単語全体$\}$
- ゲノム解析
  - ゲノム： $\Sigma = \{A, T, G, C\}$
  - タンパク質： $\Sigma = \{$アミノ酸$\}$（20種類）

ストリングカーネルの応用

- ゲノム配列のアラインメント
- タンパク質の構造予測
  - アミノ酸配列： Σ = 20種のアミノ酸

```
7LES_DROME      LKLLRFLGSGAFGEVYEGQLKTE....DSEEPQRVAIKSLRK.......
ABL1_CAEEL      IIMHNKLGGGQYGDVYEGYWK........RHDCTIAVKALK........
BFR2_HUMAN      LTLGKPLGEGCFGQVVMAEAVGIDK.DKPKEAVTVAVKMLKDD.....A
TRKA_HUMAN      IVLKWELGEGAFGKVFLAECHNLL...PEQDKMLVAVKALK........
```

    配列　→　立体構造のクラスを予測

  - データベース

    SCOP（Structural Classification of Proteins）など

p-スペクトラムカーネル

– 長さ p の部分列の出現回数を特徴ベクトルとする

$$|\Sigma| = m, \quad u \in \Sigma^p$$

$$\phi_u^p(s) = \left|\{(w_1, w_2) \in \Sigma^* \times \Sigma^* \mid s = w_1 u w_2\}\right| \quad \cdots \quad s \text{ の中の } u \text{ の出現回数}$$

$$\Phi : \Sigma^* \to H \cong \mathbf{R}^{m^p}, \qquad \Phi^p(s) = \left(\phi_u^p(s)\right)_{u \in \Sigma^p}$$

特徴空間: 長さ p の列全体 $\cdots$ m^p 次元

$$K_p(s,t) = \sum_{u \in \Sigma^p} \phi_u^p(s)\phi_u^p(t) \quad = \left\langle \Phi^p(s), \Phi^p(t) \right\rangle_H$$

s = “statistics”　　t = “pastapistan”

3-スペクトラム

s: sta, tat, ati, tis, ist, sti, tic, ics

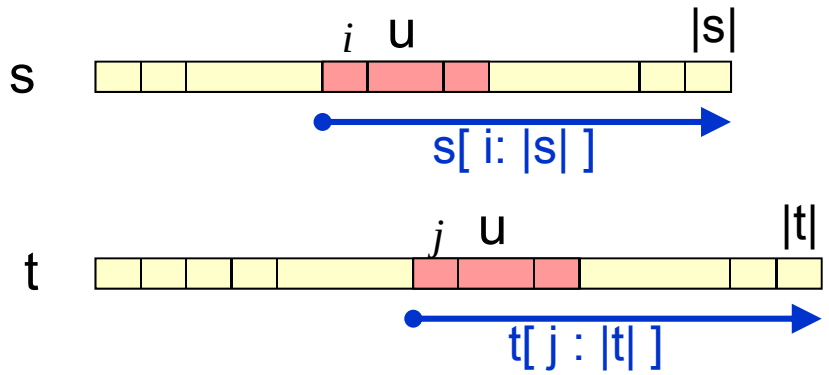
t : pas, ast, sta, tap, api, pis, ist, sta, tan

	sta	tat	ati	tis	ist	sti	tic	ics	pas	ast	tap	api	pis	tan
Φ(s)	1	1	1	1	1	1	1	1	0	0	0	0	0	0
Φ(t)	2	0	0	0	1	0	0	0	1	1	1	1	1	1

$K_3(s, t) = 1 \cdot 2 + 1 \cdot 1 = 3$

■ p-スペクトラムカーネルの計算法

$$K_p(s,t) = \sum_{u \in \Sigma^p} \phi_u^p(s)\phi_u^p(t) \quad = \left\langle \Phi^p(s), \Phi^p(t) \right\rangle_H$$



– 直接的な計算
部分列 u を，途中から始まる
部分列 suffix（接尾辞）の
先頭（prefix)と思う

$$h_u^p(s,t) = \begin{cases} 1 & \text{s の } p\text{-prefix} = \text{t の } p\text{-prefix} \\ 0 & \text{s の } p\text{-prefix} \neq \text{t の } p\text{-prefix} \end{cases}$$

⇨

$$K_p(s,t) = \sum_{i=1}^{|s|-p+1} \sum_{j=1}^{|t|-p+1} h_p(s[i:i+p-1], t[j:j+p-1])$$

計算量 ＝ O(p| s || t |)

■ p-スペクトラムカーネルの計算法(II)

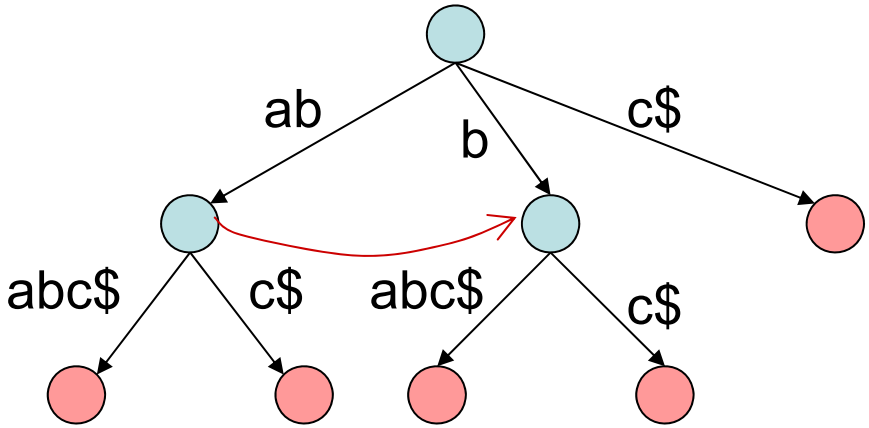
実は，|s|+|t| に対して線形時間 O(p(| s | + | t |)) で計算する方法がある

– Suffix Tree

ストリングのすべての suffix を木構造で効率的に表すアルゴリズム

例）ababc

ababc
babc
abc
bc
c



– 詳しくは Vishwanathan & Smola 03, Gusfield 97.

他のストリングカーネル

- より複雑な部分列を用いる
  - All-subsequence kernel： すべての部分列を比べる
  - Gap weighted kernel: ギャップを許す
  - Mismatch kernel: Leslie et al. (2003)

  計算量は大きくなる

- 確率的な考えによるもの
  - Fisher kernel: Jaakkola & Haussler (1999) HMMでモデル化し，分布間のdivergenceをはかる

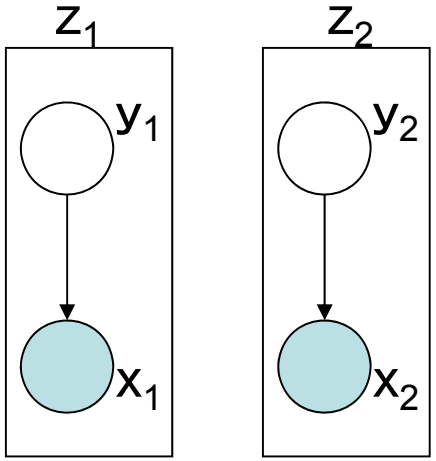
Marginalized kernel

■ 確率モデルにもとづくカーネル設計

$z = (x, y)$

x：観測される変数

y：観測されない隠れ変数
（データを生成する構造）



$p(x,y)$ ： (x, y) に対する確率モデル

$k_z(z_1, z_2)$ ： z に対する正定値カーネル

y_1, y_2 の状態全体

$$k(x_1, x_2) = \sum_{y_1} \sum_{y_2} p(y_1 \mid x_1)\, p(y_2 \mid x_2)\, k_z((x_1, y_1), (x_2, y_2))$$

■ 例

unknown

y_1 1 2 2 1 2 2 1 2 2

x_1 **A C G G T T C A A**

known

unknown

y_2 1 2 2 1 2 2 1

x_2 **A C C G T A C**

known

exon / intron

DNA

– $p(x, y)$ は隠れマルコフモデル（HMM）によって記述済み（y: 隠れ状態）

– $$k_z(z_1, z_2) = \frac{1}{|z_1||z_2|} \sum_{i=1,2} \sum_{a \in \{A,T,G,C\}} C_{ai}(z_1) C_{ai}(z_2)$$

$C_{ai}(z)$: (a, i) のカウント

1 A	1 C	1 G	1 T	2 A	2 C	2 G	2 T
1	1	1	0	2	1	1	2

– Marginalized kernel

$$k(x_1, x_2) = \sum_{y_1} \sum_{y_2} p(y_1 \mid x_1) p(y_2 \mid x_2) k_z(z_1, z_2)$$

HMMから計算

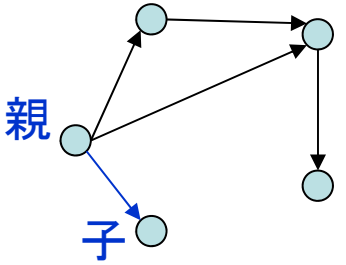
1 A	1 C	1 G	1 T	2 A	2 C	2 G	2 T
1	1	1	0	1	2	0	1

グラフとツリー

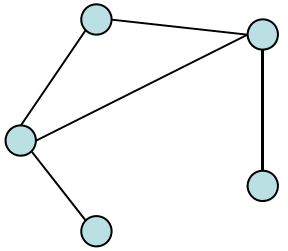
■ グラフ

- V： ノード（node, vertex）・・・ 有限集合
- E： エッジ（edge）・・・ V x V の部分集合
- 有向グラフ： E の向きを考えたもの

  (a, b) ∈ E のとき，aからbへ矢印を描く
  - ノード a の親： (b,a) ∈E なる b
  - ノード a の子： (a,b) ∈E なる b
- 無向グラフ： E の向きを忘れたもの

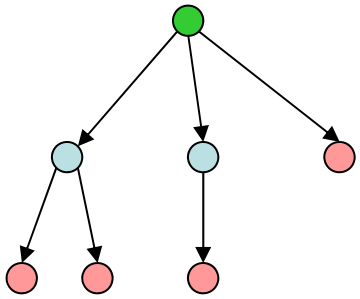


有向グラフ


無向グラフ

■ ツリー（directed rooted tree）

- 連結した有向グラフで，親の無いルートノードが存在し，他の各ノードは親を1個だけ持つもの
- リーフ：ツリーの中で子の無いノード


ツリー

ツリーカーネル

– ツリー全体の集合上に定義された正定値カーネル

Φ : ツリー T α $\Phi(T)$ $\in$ H 特徴空間（ベクトル空間）

– 代表的な例

サブツリーの一致によりカーネルを定義する

- All-subtrees kernel

$$k(T_1, T_2) = \sum_{S:\ \text{ツリー}} \phi_S(T_1)\phi_S(T_2)$$

$$\phi_S(T) = \begin{cases} 1 & T \text{ が } S \text{ をサブツリーとして含む} \\ 0 & T \text{ が } S \text{ をサブツリーとして含まない} \end{cases}$$

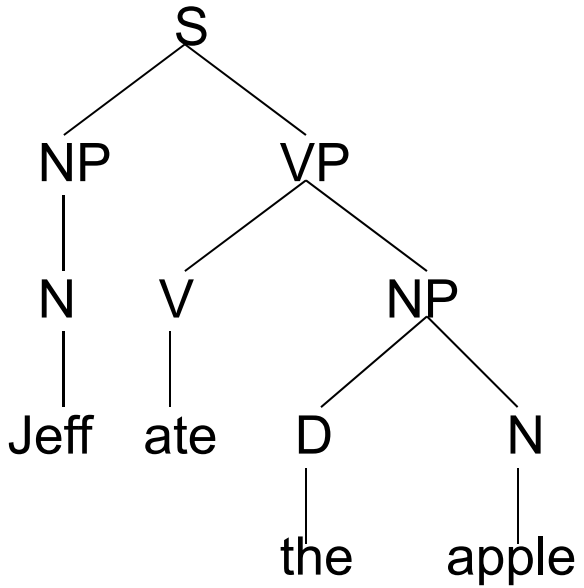
- 再帰式で計算可能. 計算量 ＝ $O(|T_1|\,|T_2|)$

– 詳細は Collins & Duffy (2002, NIPS) などを参照

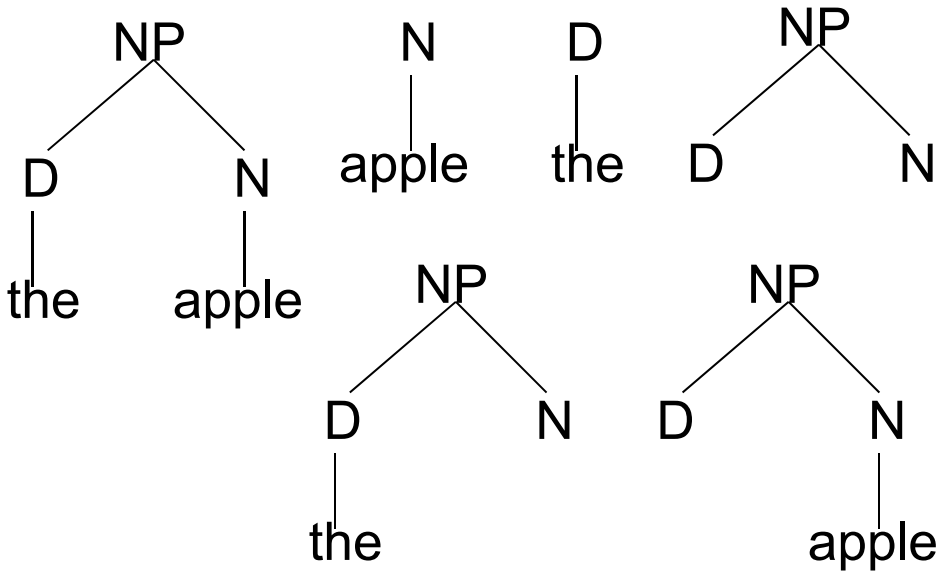
– 自然言語処理への応用

構文解析

Jeff ate the apple.



サブツリーの例



グラフカーネル

■ グラフ上に定義された正定値カーネル

– グラフとグラフの類似度を測る.

– ラベル付グラフ

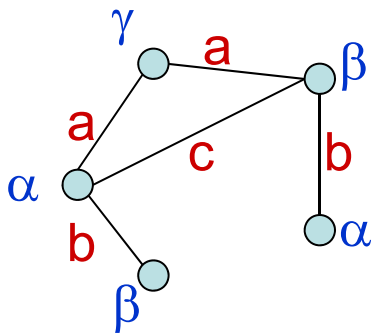
ノードとエッジにラベルがついている.

L: ラベルの集合（有限集合）

ラベル付グラフ　G = (V, E, h)

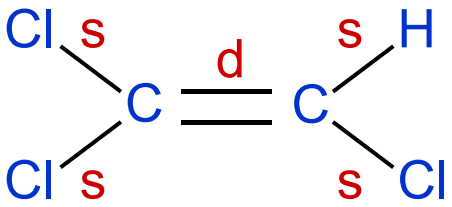
V: ノード, E:エッジ,

h : V∪E → L　ラベル付けの写像



– 応用

- 化合物の毒性予測
- 自然言語処理



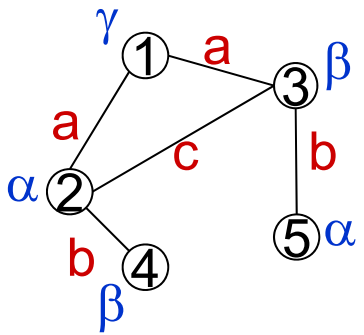
■ Marginalized graph kernel

– 系列のラベル

s: $v_1\ v_2\ v_3\ v_5\ v_3\ \cdots$

→ $H(s) = h(v_1)h(e_{12})h(v_2)h(e_{23})h(v_3)h(v_{35})h(v_5)\cdots$

$= \gamma\ a\ \alpha\ c\ \beta\ b\ \alpha\ b\ \beta\ \cdots$



– 系列の確率 – ランダムウォーク

- ノード間の遷移確率

$$p(v_j \mid v_i) = \begin{cases} 1/(i\text{の隣接ノードの数}) & (i,j) \in E \\ 0 & (i,j) \notin E \end{cases}$$

- 系列の確率

$$p(s) = p(v_1)p(v_2 \mid v_1)p(v_3 \mid v_2)p(v_5 \mid v_3)p(v_5 \mid v_3)\Lambda$$

グラフ上のランダムウォークにより生じる系列の確率

– ラベル系列に対するカーネル

$$K_L : L^* \times L^*, \qquad K_L(H_1, H_2) = \begin{cases} 1 & (H_1 = H_2) \\ 0 & (H_1 \neq H_2) \end{cases}$$

– Marginalized graph kernel

$G_1 = (V_1, E_1, h_1), \quad G_2 = (V_2, E_2, h_2)$

$$K(G_1, G_2) = \sum_{\substack{s \in V_1^* \\ t \in V_2^*}} p_1(s)\, p_2(t) K_L(H_1(s), H_2(t))$$

H_1, H_2 : それぞれ h_1, h_2 から決まるラベル関数
V_1^*, V_2^* : それぞれ V_1, V_2 をアルファベットとする系列全体

- ランダムウォークにおいて，同じパスが生じる確率
- Marginalized kernel のひとつとみなせる

– 詳しくは，Kashima et al. (2003), Mahé, et al. (2004)

構造化データ上のカーネルの問題点

■ 計算量

– k(x,y) の計算にかかる時間

O(|s| |t|) でも，サイズが大きくなると困難

– データ数

グラム行列の計算は （データ数）^{2} のオーダー

– SCOPデータベース： 配列の長さ～数百， 配列データの数～数千

References

福水 「カーネル法入門」 3章 朝倉書店 2010

Williams, C. K. I. and M. Seeger. (2001) Using the Nyström method to speed up kernel machines. *Advances in Neural Information Processing Systems*, 13:682–688.

Fine, S. and K. Scheinberg. (2001) Efficient SVM Training Using Low-Rank Kernel Representations. *Journal of Machine Learning Research*, 2:243-264.

Widom, H. (1963) Asymptotic behavior of the eigenvalues of certain integral equations. *Transactions of the American Mathematical Society*, 109:278{295, 1963.

Widom, H. (1964) Asymptotic behavior of the eigenvalues of certain integral equations II. *Archive for Rational Mechanics and Analysis*, 17:215{229, 1964.

Rahimi A. and Recht B. (2008) Random Features for Large-Scale Kernel Machines. Advances in Neural Information Processing Systems 20, 1177-1184.

Tsang, I.W., J.T. Kwok, and Pak-Ming Cheung (2005) Core Vector Machines: Fast SVM Training on Very Large Data Sets. *Journal of Machine Learning Research* 6 363–392.

Lodhi, H., C. Saunders, J. Shawe-Taylor, N. Cristianini, C. Watkins. (2002) Text Classification using String Kernels. J. Machine Learning Research, 2 (Feb): 419-444.

Leslie, C., E. Eskin, A. Cohen, J. Weston and W. S. Noble. (2003) Mismatch string kernels for SVM protein classification. Advances in Neural Information Processing Systems 15, pp. 1441-1448.

Rousu, J., and J. Shawe-Taylor. (2004) Efficient computation of gap-weighted string kernels on large alphabets. Proc. PASCAL Workshop Learning Methods for Text Understanding and Mining.

Dan Gusfield. Algorithms on Strings, Trees, and Sequences. Cambridge Univ. Press. 1997.

Jaakkola, T.S. and D. Haussler. (1999) Exploiting generative models in discriminative classifiers. Advances in neural information processing systems 11. pp.487-493.

Collins, M. & N. Duffy. (2002) Convolution Kernels for Natural Language. Advances in Neural Information Processing Systems 14.

Tsuda, K., T. Kin, and K. Asai. (2002) Marginalized kernels for biological sequences. Bioinformatics, 18. S268-S275.

Kashima, H., K. Tsuda and A. Inokuchi. (2003) Marginalized Kernels Between Labeled Graphs. Proc. 20th Intern. Conf. Machine Learning (ICML2003).

Mahé, P., N. Ueda, T. Akutsu, J.-L. Perret and J.-P. Vert. (2004) Extensions of marginalized graph kernels. Proc. 21th Intern. Conf. Machine Learning (ICML 2004), p.552-559.

Schlkopf, B., K. Tsuda, J-P. Vert (Editor) Kernel Methods in Computational Biology. Bradford Books. 2004.